Retrofitting Decision Tree Classifiers Using Kernel Density Estimation

Padhraic Smyth, Alexander Gray, Usama M. Fayyad
Jet Propulsion Laboratory M/S 525-3660
California Institute of Technology
4800 Oak Grove Drive
Pasadena, CA 9110!)-8099
{pjs,agray,fayyad}@aig.jpl.nasa.gov

Abstract

A novel method for combining decision trees and kernel density estimators is proposed. Standard classification]) trees, or class probability trees, provide piecewise constant estimates of class posterior probabilities. Kernel density estimators can provide smooth non-parametric estimates of class probabilities, but scale poorly as the dimensionality of the problem increases. This paper discusses a hybrid scheme which uses decision trees to find the relevant structure in high-dimensional classification problems and then uses local kernel density estimates to fit smooth probability estimates within this structure. Experimental results on simulated data indicate that the method provides substantial improvement over trees or density methods alone for certain classes of problems. The paper briefly discusses various extensions of the basic approach and the types of application for which the method is Lest suited.

1 INTRODUCTION

There has been considerable interest in recent years in the use of decision trees for classification and regression problems. Decision tree design algorithms have been developed in both the statistical and machine learning communities (Breiman et al. 1984; Quinlan 1992) and have successfully competed with alternative non-parametric modelling techniques (such as feedforward neural networks).

A key advantage of the decision tree approach over competing models is the understandability of the model. A decision tree using univariate node-splits is relatively easier to comprehend than models such as neural networks. This understandability is a major contributor to the widespread use of decision trees in both the machine learning and applied statistics communities, rather than any inherent capability of the decision tree model to outperform other prediction models. In fact, decision tree models can often be slightly less accurate than competing models in terms of prediction (since the functional form of the model is severely constrained) and yet be preferred as the model of choice for a particular application because of the explicit nature of the model. This is the starting point for the work in this paper. Given that there are a variety of well-established decision tree learning algorithms such as CART and C4 in widespread use, the idea of using locally flexible prediction embedded within the overall tree structure to improve the local prediction accuracy of the model is explored. In particular, we investigate the use of kernel density estimation techniques to improve the class probability prediction capabilities of existing decision trees: hence, the "retrofitting" in the title of the paper.

In certain classification applications it is often important that, given the input feature data, the classifier produce accurate estimates of posterior class probabilities, rather than simply the label of the most likely class. In speech recognition for example, the classification component may be embedded within a larger context model (typically a hidden Markov model) which uses the local classification probabilities to infer the most likely sequence of states. More generally, posterior probabilities are useful in applications such as medical diagnosis where a decisions involving unequal misclassification costs must be made. It is often the case that these costs are not known precisely in advance or may change over time. In such cases the best the classifier designer can do is provide the decision maker with estimates of class probabilities.

The standard approach to producing accurate posterior class probabilities from classification trees is known as *class probability trees*: one counts the proportions from each class which are present at the leaf nodes, based on the training data, and generates a local maximum likelihood estimate (or perhaps a smoothed variant) of the posterior class probabilities. The goal of this paper is to snow that these conventional estimates can be improved upon by combining kernel density estimation methods with decision trees.

The paper begins by reviewing the basic concepts of kernel density estimation, focusing in particular on the limitations of the method when applied to multivariate classification. An algorithm is described for combining density estimation with classification trees. Experimental results on synthetic data are discussed: the hybrid dcllsity-tree approach is shown to provide significantly better probability estimation performance than either the class probability tree or the kernel density methods on their own. Furthermore, analyzing the class probability estimation problem from a kernel density viewpoint can provide some interesting insights into estimation aspects of decision tree design. Various extensions (such as Bayesian and/or option trees) are briefly discussed and links to other dimension-reduction techniques combined with density estimation are mentioned.

The focus of this paper is on the case of numeric (real-valued) attributes or features, rather than the categorical or discrete case: density estimation techniques are much more relevant for numeric data, The methods proposed in the paper can be directly extended to handle mixed discrete/categorical/numeric data. In addition, the focus of this paper is on the tree-retrofitting problem: adding density estimates to a classification tree which was designed in a standard manner. There are obvious extensions of density estimation to the *design (or estimation)* phase of tree-1.mi]ding: these are briefly discussed where appropriate but arc not the focus of the present paper.

2 A Ii,ItV.112 W OF KERNEL DENSITY ESTIMATION

Non-parametric probability density estimation techniques have been studied in statistics since the late 1 950's. Texts by Hand (1 982), Silverman (1 986) and Scott (1 992) all provide excellent overviews of density estimation with emphasis on both theory and application. Izenmann (I 991) provides a thorough overview of recent progress on theoretical aspects of density estimation.

Kernel-based density estimation is the most widely practiced density estimation technique. Consider the univariate case of estimating the density $f(x)$ given samples $\{x_i\}$, $1 \leq i \leq N$, where $p(X < t) = \int_{-\infty}^{t} f(x)dx$ and $\int_{-\infty}^{\infty} f(x)dx = 1$ (X is a 1-dimensional random variable, $x \in [-\infty, \infty]$ denotes values of X). The idea is quite simple: one obtains an estimate $\hat{f}(x)$ by summing the contributions of the kernel $K(x - x_i)$ *over* all the samples and normalizing such that the estimate is itself a density, i.e.,

$$\hat{f}(x) = \frac{1}{Nh} \sum_{i=1}^{N} K\left(\frac{x - x_i}{h}\right) \qquad (1)$$

where h is the *bandwidth* of the estimator. $\hat{f}(x)$ directly inherits the properties of K(f), hence it is common to choose the kernel itself as a probability density fund ion. A common choice is the Gaussian kernel, K(f) $= \frac{1}{\sqrt{2\pi}} e^{(1/2)t^2}$. The Cauchy kernel is defined as $K(t) = \frac{\alpha}{1+t^2}$ where α is a scaling factor.

A point x which is close to many data points x_i will receive significant contributions from the kernels associat ccl with these data points and thus the density estimate $\hat{f}(x)$ will be large. A point x which is far away from any points x_i will only receive contributions from the tails of the associated kernels and $\hat{f}(x)$ will be relatively small. Although this idea is quite simple, it is also quite powerful: it can be shown that provided the kernel function itself obeys certain smoothness properties and the bandwidth h is chosen appropriately, asymptotically as the number of data points goes to infinity, the estimator $f(x)$ will converge to the true density $f(x)$ (Hand 1982; Silverman 1986). The optimal choice of h, given a fixed number of data points N and a particular kernel function K(.), depends on the true density function $f(x)$ but since $f(x)$ is unknown (that is the object of the exercise) one must SO1I1C']1OW find the "best" bandwidth h from the data. If h is chosen to be too small then the estimate $\hat{f}(x)$ approaches a set of delta functions about each point and the *variance* of the estimate is too high. Conversely if h is chosen too large, $\hat{f}(x)$ approaches the shape of the kernel itself and effectively ignores the data: the *bias* of the estimate is too large. The Appendix describes a widely used cross-validation scheme for finding a bandwidth value h from the data: this is a standard method in applied statistics for density estimation and is the scheme used for all of the results described in this paper.

For the multi-dimensional case the *product* kernel is commonly Used:

$$\hat{f}(x) = \frac{1}{N} \sum_{i=1}^{N} K(\underline{x}, \underline{x}_i, \underline{h}) \qquad (2)$$

where

$$K(\underline{x}, \underline{x}_i, \underline{h}) = \frac{1}{h_1 \ldots h_d} \prod_{k=1}^{d} K_k\left(\frac{x^k - x_i^k}{h_k}\right) \qquad (3)$$

and d is the number of dimensions, x^k denotes the component in dimension k of vector $\underline{x}$, K_k is the 1-dimensional kernel for the kth dimension, and the h_k represent different 1 bandwidths in each dimension. Thus the product kernel consists of the product of one-dimensional kernels: typically in practice the same kernel function is used in each dimension, i.e., $K_k(.) =$

K(.), but the bandwidths are allowed to differ. The alternative to the product kernel would be to use a full multivariate kernel in Equation (2): perhaps a Gaussian kernel with a full covariance matrix. This method has not been found very effective in practice due to the fact that the $d(d+1)/2$ bandwidth parameters for a symmetric matrix must be estimated: the product kernel only requires the estimates of d bandwidths and is widely recommended in the literature and typically used in practical situations. Although the product kernel uses *kernel independence* in Equation (3) this does not imply that any form of *attribute independence* is being assumed: in fact, as in the 1-dimensional case, it can be shown that the product kernel estimate asymptotically approaches the true density as the sample size N increases, under the appropriate assumptions (Cacoullos 1966).

3 CLASSIFICATION WITH KERNEL DENSITY ESTIMATES

Kernel density estimation can be used as the basis for a classification method as follows. Consider that there are m classes, $\omega_1, \ldots, \omega_m$ and denote the d-dimensional attribute/feature vector as $\underline{x}$. As usual, for classification problems, there is a set of training data available where for each sample $\underline{x}_i$, the true class label is known. For each class ω_j, take only the training data that belongs to class j and estimate $\hat{f}_j(\underline{x}) = \hat{f}(\underline{x}|\omega_j)$ which is the density estimate for the data from that class (in isolation, derived independently from the other classes). $\hat{f}_j(\underline{x})$ can be estimated using the methods described in the last section. Bayes' rule is then used for classification:

$$\hat{p}(\omega_j|\underline{x}) = \frac{\hat{f}_j(\underline{x})p(\omega_j)}{\sum_{i=1}^{m} \hat{f}_i(\underline{x})p(\omega_i)}, \qquad 1 \leq j \leq m, \qquad (4)$$

where the prior or marginal probabilities of each class, $p(\omega_i)$, are estimated from the data in the usual fashion.

This classification method has existed since the 1950's (often referred to as "Parzen windows") but has not seen widespread practical use. One reason for its limited application in practice has been the computational complexity of the method: all of the data must be stored and all the kernel contributions summed to make a classification estimate. However, with modern computation and memory capabilities this need not be much of a problem except for very large data sets.

A more fundamental problem is the fact that density estimation tends to scale poorly as the dimensionality d of the problem increases. In particular, it can be shown theoretically (Scott 1992) that to achieve a constant approximation error as the number of dimensions grows one needs exponentially many more examples. Thus, in practice, density estimation techniques are rarely used directly for high-dimensional problems. Naturally, for classification tasks, this is a significant drawback since often there may be a large number of attributes only some of which are relevant. Because the kernel classification method builds a density model for the data using *all* of the input dimensions it maybe relatively inefficient in its use of the data compared to a discriminative method such as a decision tree which constructs a model using only those dimensions which are necessary to discriminate between classes. Thus, given the desirable probability estimation properties of kernel methods, one is motivated to seek hybrid kernel methods which only use the relevant discriminative dimensions.

4 DECISION TREE DENSITY ESTIMATORS

The key idea we introduce in this paper is as follows: at each node in the tree, estimate the posterior class probabilities (given the input data) using a multivariate product kernel density estimator, where the density estimator only uses those input features which have been used in the branch partitions leading to that node. Intuitively the method tries to combine the better aspects of both trees and density estimators. The motivation is two-fold:

1. **Probability Estimation:**
   improve the class probability estimation properties of decision trees. Trees provide piecewise constant probability estimates which are non-smooth as one crosses decision boundaries, i.e., one will tend to get very different class probability estimates by descending on either side of a node with threshold t. Furthermore, the class probability estimates will be the same for all for inputs $\underline{x}$ which fall into a particular leaf (or internal node): the exact value of $\underline{x}$ is not used in determining the posterior probabilities. For problems with a fair degree of uncertainty (the Bayes error rate for the problem is relatively high) it is certainly reasonable to expect that the class probabilities may vary considerably within a particular leaf or node, e.g., from $p(\omega_i|\underline{x}) \approx 0.5$ near the split to $p(\omega_i|\underline{x}) \approx 1.0$ far away from the split. The kernel addition proposed here replaces the non-smooth, piecewise constant probability estimates at each leaf, with a smooth, non-parametric, kernel based estimate of the posterior probability function.

2. **Problem Dimensionality:**
   reduce the number of variables which must be included in the multivariate kernel density estimate by using the information provided by the decision tree structure, As discussed earlier, kernel methods will fail on high dimensional problems. The hybrid method seeks to identify the discriminative dimensions via the tree structure and then uses those dimensions to construct local density

estimates.

The proposed method (details of which are provided in the next section) can be viewed as either a method for fitting better probability estimates to trees, or a way to construct, kernel classifiers in high dimensions using local discriminative information. In terms of decision trees, the method in general is applicable to both (1) tree *design* and *(2) prediction* using a particular tree: the latter aspect can be considered "retrofitting" an existing tree structure with a density estimator. In this paper we will only consider the "retrofitting" aspect of the problem there are several interesting avenues to explore in terms of tree design combined with density estimation, but these are not pursued in detail here.

The hybrid density-tree idea is well-suited to certain kinds of problems. In particular it is suited to high-dimensional problems where accurate class probability estimates are desirable and the Bayes error rate is not too low. If the Bayes error rate for the problem is very low, then all of the posterior class probabilities will be close to 1 or O and there is little advantage to using a kernel density estimator and a standard decision tree classifier should be preferred (the piecewise constant, estimates of the trees will work fine). Similarly, if the problem is low-dimensional, then the kernel density estimator can be used directly.

5 DETAILS ON DECISION TREE DENSITY ESTIMATORS

The basic tree-density algorithm for the results described in this paper operates as follows:

1. **Density Estimation:**
   Run a kernel density bandwidth estimation method on the training data (such as that described in the Appendix) to select bandwidths h_k, $1 \leq k \leq d$, for each of the input dimensions and for each of the classes $\omega_j, 1 \leq j \leq m$.
2. **Decision Tree Design:**
   Generate a classification tree from the training data using a standard decision tree design algorithm, e.g., CART, C4, etc. If pruning is part of the basic algorithm (as in CART) then produce a pruned tree as the final result.
3. **Retrofitting the Decision Tree for Prediction:**
   To perform class probability prediction on a new data point $\underline{x}$:

   3.1 Pass the test data point down the tree in the usual manner to a leaf node.

   3.2 Generate a local density estimate for each class as follows:

$$\hat{f}(\underline{x}|\omega_j) = \frac{1}{N_j}\sum_{i=1}^{N_j}\prod_{k\in\text{path}}\frac{1}{h_k}K\left(\frac{x^k - x_i^k}{h_k}\right) \qquad (5)$$

   where $k \in \text{path}$ denotes that the product is taken only over those attributes which appear in tests on the path from the root to that particular leaf, N_j is the number of training data points which belong to class ω_j, and the sum $\sum_{i=1}^{N_j}$ is taken to be over only training data points belonging to class ω_j.

   3.3 Estimate the class probabilities, $p(\omega_j|\underline{x})$, using the density estimates from Equation (5) combined with Bayes' rule (Equation (4)).

Many variations on this basic theme exist. For example, the density estimates could also be used as part of the tree design phase. Bayesian averaging over option trees or smoothing over internal nodes could also be incorporated directly. Alternative density estimation methods are possible, such as locally adaptive methods or kernel techniques which avoid Bayes' rule and seek to estimate $p(\omega_j|\underline{x})$ directly (Lauder 1983) but still use the information in the tree structure.

For the purposes of this paper we have restricted our attention to the simple method described above in order to evaluate the potential utility of the overall idea.

6 EXPERIMENTAL RESULTS

6.1 EXPERIMENTAL DATASETS

In terms of probability estimation, the class probabilities $p(\omega_j|\underline{x}_i)$, where $\underline{x}_i$, $1 \leq i \leq N$, is a datum from the training data set, are typically not known for real-world training data sets: all one typically knows are the class-labels but not the posterior probabilities given $\underline{x}_i$. Thus, to accurately assess the performance of a class probability estimator one needs to use simulated data for which the true posterior probabilities are known. (Note that an alternative approach is to estimate the difference between the probability estimates and the true probabilities via the half-Brier score (Buntine and Caruana 1992), which essentially substitutes " 1" or " O" for the true probability depending on which class is true however, this can be an inaccurate estimate when the sample size is small and the probabilities themselves are not near 0 or 1).

We chose some deceptively simple simulated problems to test the methodology: variants of a 2-class problem where the data for each class are distributed in a Gaussian manner with 12 dimensions. The two classes differ only in 1 or 2 dimensions depending on the problem: thus, from a discrimination/classification point of view there are 11 or 10 irrelevant noise dimensions.

- Problem 1: The two classes only differ in 1 dimension, $\mu_1 = 0$, $\mu_2 = 1$, $\sigma_1 = \sigma_2 = 1$: thus, there is significant overlap in this dimension. Both classes are equally likely. The Bayes error rate (the minimum achievable error rate for the problem) is about 0.31. The other 11 dimensions are independent and consist of zero-mean unit-variance Gaussian noise. The optimal decision rule for the problem consists of a single split along the first dimension.

- Problem 2: This is the same as Problem 1 except that the mean of the second class is now $\mu_2 = (\sqrt{2}/2, \sqrt{2}/2)$ in the first two dimensions and the covariance matrix in the first two dimensions is $0.5I$ where I is the identity matrix. The mean for all dimensions (except the first two dimensions of class 2) is zero: so the other 10 dimensions are irrelevant. The optimal decision boundary for this problem is only a function of the first two dimensions but is quadratic rather than linear. The Bayes error rate is about 0.23.
- Problem 3: Class 1 is distributed in the same manner as in problems 1 and 2, but class 2 is now a mixture of 2 components in the first 2 dimensions: one is centred at $(-\sqrt{2}, -\sqrt{2})$, the other at $(\sqrt{2}, \sqrt{2})$ and each component has covariance matrix of $0.5I$. Class one is defined to have a prior probability of 1/3 and class 2 2/3 for this problem.. Once again the mean for all dimensions (except the first two dimensions of class 2) is zero: so the other 10 dimensions are irrelevant and the optimal decision boundaries are a nonlinear function of the first two dimensions for the problem. The Bayes error rate for this problem is estimated to be about 0.14.

Several other simulated problems were used to test the methodology but are not reported here - all were variants of low-dimensional Gaussian or mixture of Gaussians embedded in a higher dimensional space. In all experiments the results were qualitatively the same as those described below.

6.2 EXPERIMENTAL METHODOLOGY

We monitored both the classification error rate and the probability estimation error for a variety of classifiers as a function of sample size. We varied sample training sizes from S to 2048. For a given sample size, 20 independent training sets were generated according to the probability models described above (for Problems (1), (2) and (3)). Each classifier was trained on each of the 20 independent training datasets. The error rate of each Classifier, for a given training data set for a particular sample size, was evaluated empirically on an independent test set of 3000 samples. The *mean* error rate of a particular classifier over the 20 training data sets was then calculated, along with the standard deviation. our experimental results are thus in the form of mean error rates for a given classifier as a function of sample size. The standard deviations of the means are not shown on the graphs to reduce clutter.

Calculation of *classification error* rate on the test set was carried out in the standard manner. calculation of the *estimation error for class probabilities* typically can be carried out using a variety of methods. We chose to use the mean absolute distance:

$$E = \frac{1}{N_{test}} \sum_{i=1}^{N_{test}} \frac{1}{m} \sum_{j=1}^{m} \left| p(\omega_j|\underline{x}_i) - \hat{p}(\omega_j|\underline{x}_i) \right|, \qquad (6)$$

where N_{test} is the number of test data points.

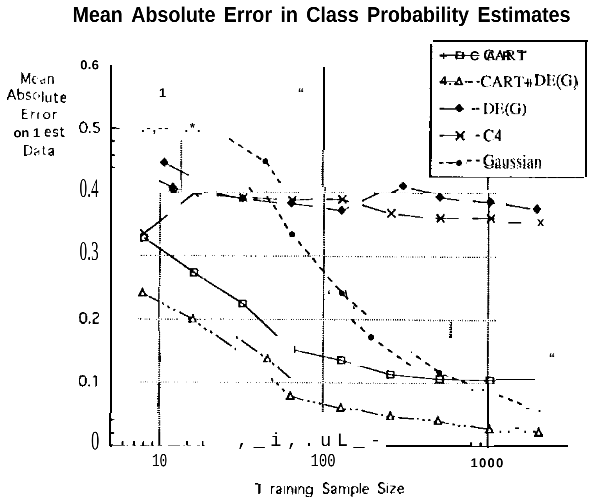


Figure 1: Mean absolute error for class probabilities as a function of training sample size for Problem 1

6.3 CLASSIFIERS USED

For our standard decision tree classifiers we used both the CART and C4 algorithms as implemented in the IND software package (Buntine and Caruana 1992), using default settings. For density estimation we used the product kernel density method described in Sections 2 and 3 (and cross-validation method as in the Appendix). We experimented with both Gaussian and Cauchy kernel shapes (Silverman 1986) to get a rough idea of the sensitivity of the method to kernel shape. We also included a maximum-likelihood Gaussian classifier using separate full covariance matrices which are estimated from the data for each class.

Other decision tree methods were experimented with, such as ID3. In general we found that trees that did not use pruning or cross-validation were unable to find the relevant dimensions for the problem and, thus, the results are not shown on the plots.

6.4 DISCUSSION OF EXPERIMENTAL RESULTS

Figures 1 and 2 show the probability estimation error and the classification error rate, respectively, as a function of sample size for Problem 1. Both figures

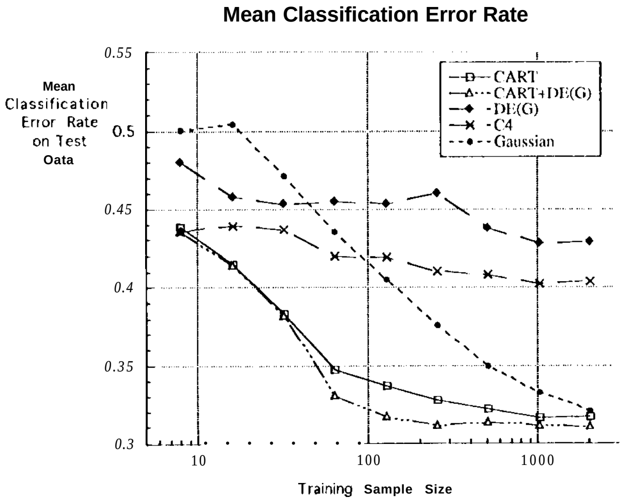


Figure 2: Mean classification error as a function of training sample size for Problem 1

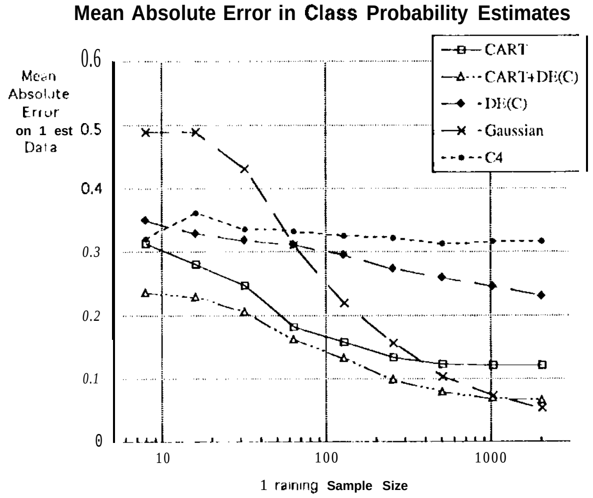


Figure 3: Mean absolute error for class probabilities as a function of training sample size for Problem 2

clearly demonstrate the benefit of using only the relevant dimensions: the full Gaussian and density estimation models (Gaussian and DE, respectively) converge slowly to the optimal error rates, while the methods which try to select the relevant dimensions (CART and CART+DE(G)) are substantially more accurate. The "G" in "CART+DE(G)" and "DE(G)" refers to the fact that for this problem the results are shown for the density estimation method using the Gaussian kernel. CART+DE is significantly better than CART alone in terms of probability approximation (Figure 1 (a)) as one might expect.

Note that CART dots not converge to the optimal asymptotic error of zero as the sample size increases due to its piecewise constant probability estimation function which acts as a llol]-zero bias term independent of the sample size. It is also worthy of note that for these data sets, CART performs significantly better than C4. We suspect that the reason for this is that the pruning methods used in CART happen to be more appropriate for these problems where the optimal decision tree solution consists of a very small decision tree. In order to avoid clutter in the presentation of the results, we show the results of the tree+-density method only for CART. We provide the C4 curve just as reference baseline for how another tree algorithm performs.

In terms of classification accuracy (Figure 2), CART+DE appears slightly more accurate than CART although this difference is probably not significant. This is not surprising since one would expect on average that if a model produces more accurate class probability estimates that it will also be more accurate in its classifications although clearly this need not always be true since the minimum error rate classifier need only know where the optimal decision boundaries are located irrespective of the values of the class probabilities.

The curve for density estimation in 12 dimensions (labeled DE(G) or DE(C) in the figures) shows how density estimation benefits from the dimensionality reduction provided by the tree structure. Without the tree addition (CART+DE), the density estimation method (DE) is quite inaccurate.

The Gaussian classifier, which is asymptotically the optimal classifier for this problem, exhibits the usual $\frac{1}{N}$ scaling performance, where N is the number of training samples: note that even at 2048 samples it still has not reached the accuracy of the CART+DE method.

For 1 'roblem 2 we show the results for the Cauchy kernel (CART+ I) N(C)) to illustrate that for these problems at least the tree-l density method dots not appear over-sclmitive to the exact shape of the kernel used in the density estimation phase. Figures 3 and 4 show the probability approximation error and classification error respectively as a function of sample size. The results are qualitatively similar to those obtained for 1 'roblem 1, namely that the CART+DE method outperforms the other methods over a wide range of sample sizes: the only difference is that the Gaussian model converges more quickly as a function of sample size for this problem, relative to the others, probably due 10 the fact that 2 of 12 dimensions are now relevant rather than just one in the first problem.

Figures 5 and 6 show the corresponding results for Problem 3. 1 lere we plot only the tree and tree-l density results to clearly demonstrate the benefits of the retrofitting approach. Both CART+DE(G)

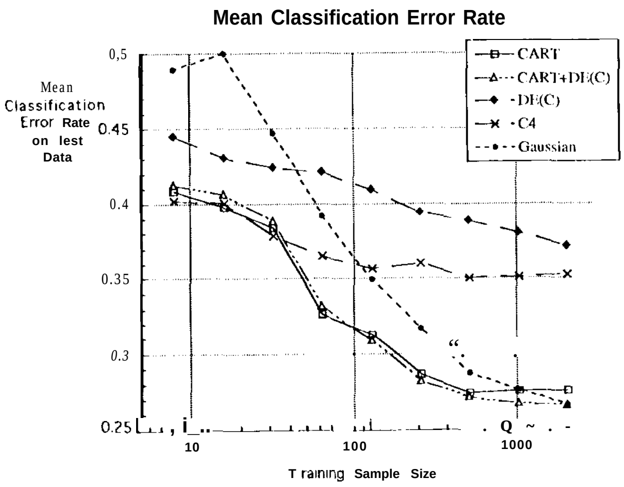


Figure 4: Mean classification error as a function of training sample size for Problem 2

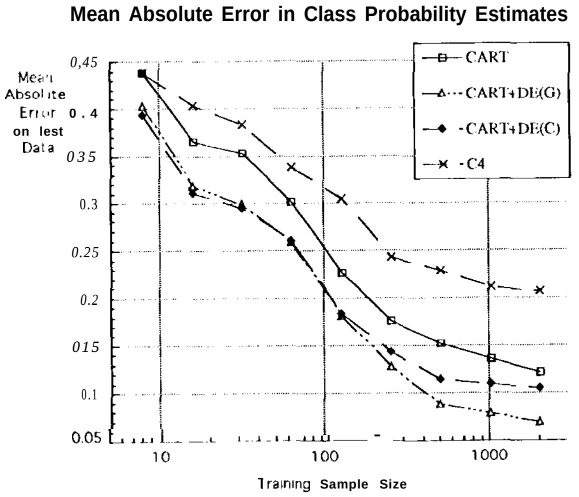


Figure 5: Mean absolute error for class probabilities as a function of training sample size for Problem 3 (mixture classification problem)

and CART+DE(C) dominate the performance of the tree methods over a wide variety of sample sizes. The Gaussian kernel method outperforms the Cauchy kernel method, probably due to the fact that underlying densities for the problem are themselves Gaussian, and CART outperforms C4 due its tendency to prune to smaller trees.

The main point to note from the experimental results in total is that the tree+density methods can provide significant improvement in terms of class probability estimation across a variety of problems and training sample sires, while the classification accuracy of the resulting tree is not affected adversely and in many cases appears to be slightly improved. The empirical results, combined with our understanding of the basic theory, indicate that the combination of robust tree algorithms and accurate density estimators can produce useful results.

7 RELATED WORK, EXTENSIONS, AND DISCUSSION

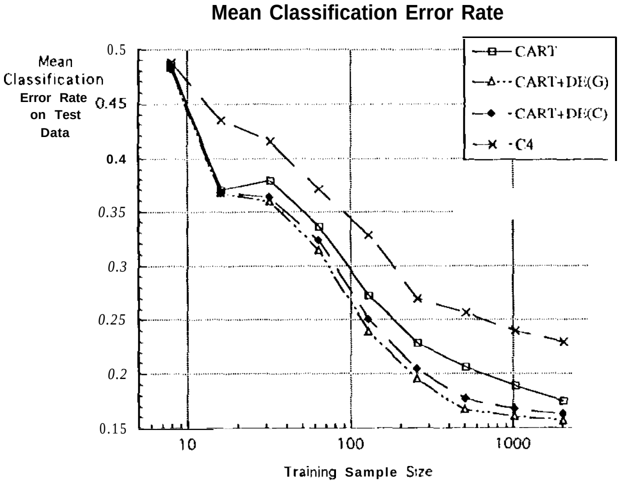
Buntine (1993) investigated a Bayesian approach to both tree design and prediction. For class probability prediction, Buntine advocates averaging the class probability estimates obtained at internal nodes in order to get the best estimate and also discusses averaging over multiple tree structures ("option trees").

Walker (1992) has investigated the following problem: using the same data to generate the class probability estimates as is used to design the tree will result in biased estimates in practice since the tree design process favors splitting the feature space into regions where the class probabilities appear to be near zero or one. Walker investigated the use of various cross-validation strategies to deal with this problem and demonstrated improved probability estimation performance compared to the standard approach.

Both of these approaches complement the general method proposed in this paper and indeed the method proposed here could likely be improved upon were it to incorporate either of the Bayesian or cross-validation strategies.

Friedman (1995) proposes sophisticated data-driven classification strategies which depend on local selection of relevant distance metrics for near neighbor type algorithms. The overall approach is partly motivated by similar concerns to those expressed in this paper, namely, that standard decision tree methods are limited to piecewise-constant class probability estimates. However, Friedman's work appears primarily motivated by a desire to improve model prediction capabilities , resulting in complex models which are essentially nearest-neighbor in form and (unlike the met hod proposed here) do not possess the understandability of the decision tree structure.

As mentioned earlier, there are a variety of potential extrusions of the basic method described here. Decision tree density estimators can in principle can be extended to regression trees, trees with multivariate splits, smoothed class probability prediction over internal nodes, averaging over multiple trees and so forth. The use of density estimation during decision tree design is also possible: for small sample sires at nodes being considered for splits, the density estimate could serve to improve the estimates of split criteria and perhaps produce more refined estimates of the location of the best split.



Figaro 6: Mean classification error as a function of training sample size for Problem 3 (mixture classification problem)

From a density estimation viewpoint, the proposed method is probably most closely related to projection pursuit density estimation (Silverman 1986): in this method, "interesting" low-dimensional projections of a high-dimensional dataset are found and density estimation is performed in this low-dimensional projection. This technique is usually carried out in the context of unsupervised learning or clustering. The proposed decision tree density estimators could be viewed as a *supervised* learning analog to the projection pui - suit methods.

A reasonable question to ask is whether one sacrifices the interpretability of a decision tree classifier using this method? This should not be the case. The structure of the tree is retained but a more complicated kernel model is used for prediction. Thus a user can still interpret the structure of the tree in terms of which variables are relevant to the classification problem, but underlying the tree structure is a more complex, memory-based, prediction scheme (which is not of direct concern to the user). Thins, for explanation purposes one can still retain the tree structure.

8 CONCLUSION

A novel method for combining decision trees ant] kernel density estimators was proposed. On simulated data sets the method was demonstrated to provide improved performance in terms of class probability estimation over either trees or density methods alone. The method is particularly useful for classification problems where class probability estimates are important, and where there is a relatively small amount of training data relative to the dimensionality of the problem (which frequently occurs in practical problems of intcrest)

Acknowledgements

The authors would like to thank Wray Buntine for making the IND software available to usc in the experiments described in the paper. The research described in this paper was performed at the Jet Propulsion Laboratory, California Institute of Technology, under a contract with the National Aeronautics and Space Administration and was supported in part by ARPA and ONR under grant number N00014-92-J-1860.

References

Breiman, 1,. Friedman, J. H., Olshen, R. A. and Stone, C. J. (1984). *Classification and Regression Trees*, Belmont, CA: Wadsworth.

Buntine, W. and Caruana, R. (1992), *An Introduction to IND and Recursive Partitioning.* Version 2.1, NASA Ames Research Center.

Buntine, W. (1993). Learning classification trees. In Artificial Inteligence Frontiers in Statistics: AI and Statistics III, London, UK: Chapman and Hall, 183 201.

Cacoullos, T. (1966). Estimation of a multivariate density. Ann. *Inst. Statist. Math.*, 18: 178-189.

Hand, D. J. (1982). *Kernel Discriminant Analysis.* Chichester, UK: Research Studies Press (John Wiley and Sons). Friedman, J. H. (1995). Flexible metric nearest-neighbor classification. Department of Statistics, Stanford University, preprint.

Izenmann, A. J. (1991). Recent developments in nonparametric density estimation. *J. Am. Stat. Ass.*, 86: 205 224.

Lauder, I. J. (1983). Direct kernel asessment of diagnostic probabilities. *Biometrika*, 70(1): 251-6.

Quinlan, J. R. (1992). *C4.5: Programs for Machine Learning*, Los Gatos, CA: Morgan Kaufmann.

Scott, D. W. (1992). *Multivariate Density Estimation: Theory, Practice, and Visualization.* New York: John Wiley and Sons.

Silverman, B. (1986). *Density Estimation for Statistics and Data Analysis.* New York: Chapman and Hall.

Walker, M. G. (1 992). Probability Estimation for Classification Trees and DNA Sequence Analysis, PhD thesis, Departments of Computer Science and Medicine, Stanford University.

Appendix: Univariate Bandwidth Selection for Kernel Density Estimation

Choice of a good bandwidth value h can be difficult. The theoretically optimal value (in terms of minimising the mean integrated square error between the estimate and the true density) is a function of the unknown density $f(\underline{x})$. Hence, in practice, various data-dependent techniques are used to estimate h from the data. Choosing h too small results in a very "spiky" estimate, while too large a value for h smooths out the details. The maximum likelihood solution for h is degenerate in the sense that choosing $h = O$ maximises the likelihood resulting in a density estimate which has delta functions at each training data point. Hence, cross-validation techniques have been widely used in practice (Silverman 1986). One such method is to maximise the "pseudo-likelihood": letting

$$f_i^*(x_i) = \frac{1}{(N-1)h} \sum_{j=1, j\neq i}^{N} K\left(\frac{x_i - x_j}{h}\right) \qquad (7)$$

the optimal cross-validation choice is

$$h_{CV} = \arg\max_h \left\{ \frac{1}{N} \sum_{i=1}^{N} \log f_i^*(x_i) \right\} \qquad (8)$$

The negative of the term in brackets can be shown to be an unbiased estimator of the expected cross-entropy between $\hat{f}(x)$ and $f(x)$. An alternative to likelihood cmss-validation} is least-squares cross-validation.

For the results reported in this paper we estimate the bandwidth in each dimension, for each class, independently. The data is initially scaled in each dimension to have unit variance and zero mean. Then, h_{CV} is found using an exhaustive grid search where the grid width is 0.01 and the search is over $h \in [0.2, 0.8]$.